



[IBM](#) : [developerWorks](#) : [Linux overview](#) : [Library - papers](#)

## How to make money with open-source software

Making money on a free lunch

[Donald K. Rosenberg](#)

Stromian Technologies

August 1999

*The software may be free, but that doesn't mean you can't make money in open source. Consultant Donald Rosenberg tells how a developer with open-source skills can choose among various business models and licenses. Developers wanting to become software vendors will have to decide how much control they want over their software, from full proprietary protection to a completely open license (GNU General Public License), or degrees in between, such as the Mozilla Public License. Donald introduces you to the key licenses and explains how they differ from one another.*

It's true that open-source software is free for the taking and that millions of copies of open-source programs, drivers, scripts, and extensions are downloaded over the Web with no money changing hands. It's also true that open source is a growing market, and that more people are making money at it. So how do *you*, a developer, go about it?

The simplest, easiest way is to keep on doing what you love: writing code. There is a growing demand for people with your open-source skills. Your chances are greater than ever of finding a job using your expertise in Linux, sendmail, Apache, Perl, Scriptics, or Python. You can be a regular employee or a contract employee, moving from job to job.

But you're probably reading this because you want more independence and would like to go into business for yourself. As an employee, you need one main thing: your expertise as an open-source expert. As an independent developer, however, you will need not only your open-source expertise, but also the ability to be your own boss, marketer, salesman, and business manager (are you sure you want two full-time jobs?). In short, you're going to have to go beyond being a developer.

### Types of businesses

One possibility would be to sell Penguin mugs, t-shirts, and such to the open-source community and to anyone else who wants them, but this business would make minimal use of your developer skills. Strategies that would call on both your open-source and business skills would be:

- **Systems integrator or Value-Added Reseller (VAR).** Any salesperson in a new business has to be an evangelist: You will have to promote both your business and open source. A systems integrator sells and installs computer systems, both hardware and software. You can set up small businesses with remotely-manageable, never-reboot Linux systems, or you can install Internet (Web and e-mail) operations in any size business. You are buying, reselling, and adding your expertise to deliver the best solution to the customer's problems. To help developers get into this business, some Linux lovers have set up a [Web site telling how to run a Linux consulting business](#).
- **Help desk or training (support).** This business is not installing new systems, but helping users keep their open-source systems running. Unlike smaller systems integrator operations, your support organization will need to hire people to man the phones (around the clock for important customers); training can be done remotely or on site. You will need to plan on pricing and procedures for sending out personnel to solve problems on site as well. Your business might contract directly with end-users, or subcontract to a systems integrator to provide the support for the integrator's customers.
- **Custom software development.** Think of yourself as a systems integrator without the hardware sales. You custom-produce applications or Web sites to solve the customer's problem. You might be working

#### Contents:

- [Types of businesses](#)
- [Listening to users](#)
- [Going to market](#)
- [Types of licenses](#)
- [GNU GPL](#)
- [BSD or X License](#)
- [Mozilla Public License](#)
- [Proprietary opportunities](#)
- [The new marketplace](#)
- [Resources](#)
- [About the author](#)

directly for the customer, or for the end-user's systems integrator. If John Ousterhout is right about the growing role of scripting in application development, there will be plenty of work for Scriptics, Perl, and Python developers. If you are careful to work strictly with open-source licenses, you will be able to re-use your work for other customers and build up a collection of solutions. If you are working with proprietary software, be sure to have your customer agree how much of the work you will be able to keep title to, so that you can use it in future projects. If nothing is said in the contract about who owns the resulting work (such as, "this is a work made for hire," meaning the customer owns it), then the presumption is that you, the developer, have title to the work, and you can then grant permission for the customer to use it in the form of a non-exclusive license.

One change that open source will make in this market is that developers will no longer be able to keep their customers captive by having closed or obscure code that no one else can understand. If you are following open-source procedures (that is, supplying unobscured source code), then you will need to rely on good service and pricing, besides good product, to retain your customers.

- **Hardware vendor.** The hardware comes with an operating system, of course, and often with additional software. Systems integrators don't make much money on hardware (they add value in understanding what the customer needs, installing it, and making it run), but notice that both resellers and manufacturers are doing well with hardware carrying open-source software. Cosmos Engineering began by selling hard drives with Linux installed and moved on to computers; VA Linux and TurboLinux (formerly Pacific HiTech) are doing well selling systems. Manufacturers of specialized hardware, such as Cobalt Networks and Whistle Communications (now part of IBM), provide a complete Internet-server installation, ready to plug in. You can use off-the-shelf hardware and open-source software to make a ready-to-run product. You make your money on the value you add in tuning everything to work together efficiently, right out of the box. Your customers will be both end-users and systems integrators (the integrators will provide any customization needed by their end-user customers). End-users could include any business, but especially ISPs.
- **Software publisher or Independent Software Vendor (ISV).** There is money to be made in distributing open-source software; the key is to add value in addition to the bare software. The most familiar example is Red Hat Software, which makes money distributing Linux in boxes while it gives it away over the Internet (and while other people sell Red Hat Linux on \$2 or \$3 disks). But having a leading brand does not mean there is no room for others. Users obviously have different preferences, and the smart publisher will serve those preferences. SuSE is doing well around the globe; Caldera has targeted the wing-tip crowd; TurboLinux has moved from a hardware vendor to a Linux publisher with its namesake distribution; and Slackware and Mandrake have loyal followings.

A software publisher can publish the work of others, or it can develop its own products. This is the high-wire act of computer marketing. Profits are higher here, but risks are greater, and problems of marketing and distribution are at least as difficult as the development problems. It is hard to get both right. If you're publishing a distribution of Linux, at least your customers know what Linux is; if you're publishing applications, the education burden is all yours.

## Listening to users

And now a word of caution: Be sure your product is something other people want. People who truly love technology--like developers--admire technical accomplishment and ingenuity for their own sake. Add to this prejudice the pride of parenthood, and you have a serious blind spot. End-users--who don't spend every waking moment thinking about cool software and who focus instead on making their own lives and work easier--do not admire technology just because it can be done. They are unlikely to understand your product unless they can relate it to their own needs. These people are your customers.

With proprietary software, you test the market by putting out your software as an executable, as time- or function-limited shareware or as freeware, and then follow up with your takers to see how they like it and are using it. Start with a few customers, and listen carefully to their reactions. Ask them what changes they would like--you may be surprised to find that they want the software to be simpler. If they want it simplified and say their friends would like it, you may have a horizontal application on your hands (or even a killer app!). But if they want customization of it, you may be headed down the road to becoming an integrator rather than an application vendor.

In these small-scale beginnings you will be more successful if you stick with a niche market, generally building

on your own expertise in that niche. Because a niche market is a small, specialized neighborhood, you can hope people will talk about your product and bring you more customers. Finally, if you did have a widely popular product, a Very Large Software Company might start to integrate your functionality into their Very Large Product. You are safer in a niche.

With an open-source product, your software is immediately available to anyone who can find it. Although feedback from the development community will give you some guidance on what end-users will want, remember that unless it's a tool, developers are not end-users. Just as with a proprietary product, you will have to promote your product to the niche users most likely to want it, and you will have to heed their suggestions about the product. A niche strategy lets you focus your resources for the best results.

## Going to market

How rapidly can you get up and going? While open-source developers like to point to shorter development times and higher quality of their software, the truth is the newly-formed companies that make the most noise are those with products that have been around the longest and have a proven market. Because the developer/sysadmin niche is probably the most heavily developed part of the open-source market, it's not surprising that the marketed products described here are tools rather than end-user applications.

Scriptics Corp. was set up to commercialize Tcl/Tk. The core product remains free; Scriptics does training, enhancements for niche markets, and other customization. In-house programmers and a useful Web site help keep future development centered on Scriptics.

Sendmail, Inc. commercializes the famous sendmail program by providing consulting, a large part of which involves installing and configuring sendmail for sysadmins.

Sleepycat Software started with a shorter time frame, and from a smaller base. Like the founders of Scriptics and Sendmail, the founders of Sleepycat originated their software as an open-source project (Berkeley DB) and then formed the firm to commercialize it. Their database is available under an open-source license to any user or vendor who agrees to pass along all the source code for the product and any modifications. Sleepycat will issue a special license to those vendors seeking to keep their modifications private. This is an OEM license; that is, Sleepycat software becomes part of the vendor's product. As copyright holder, Sleepycat can distribute its software under any license it chooses.

Digital Creations puts out the open-source Zope application server. The software is free, and there are no license fees on the derivative Web sites. The tool attracts attention to the company's Web site development business and serves as an advertisement for the technical prowess of its inventors. Users may take advantage of the free software, but Digital is betting some customers would rather not go to the effort of learning and using it, but would rather hire its creators for the job.

## Types of licenses

It's the open-source licenses that make open-source software so different from other software, and these licenses are hard for both insiders and outsiders to understand. Just remember that the license you choose will depend on what sort of business you want to run. Before we talk about specific licenses, remember this: Always put a copyright notice on your software. If you really intend to make money with it, register the copyright to increase your chances of winning any future disputes.

As copyright holder, you acquire a set of rights, any of which you can pass on (or not) to the people to whom you license your software. Proprietary software licenses, for instance, pass on very few of these rights to customers. As a copyright holder, you can even use different licenses with different types of customers. For instance, you could use the GPL for customers who agree to keep the product and changes to it open, and another license (BSD-type or X license) for customers who want to keep the source code for their changes to themselves. All three of the licenses described below are approved by the [Open Source Initiative](#) as conforming to the Open Source Definition.

- **GNU General Public License (GPL).** If you want your software to take advantage of the expertise of the programming community, and have the widest possible distribution, choose the GNU GPL. Developers will be more likely to contribute if they do not feel that their contributions are going to be taken private by someone, and you will enjoy unhindered distribution of your product. Development effort will center on you (as long as you keep in touch with your developer community). You'll make money, not on the software itself, but on supplying it in convenient, tested form, in proprietary enhancements (which must not be

compiled or statically linked with the GPL'd code), and in customization. You can feed these improvements back into the GPL'd code as the program makes progress.

- **BSD or X License.** If your end-users are vendors who intend to modify your product and sell it to their market, they may prefer this type of license, which keeps modifications private. The vendors then know they are the exclusive purveyors of their enhancements. There is, however, an incentive for the vendors to feed their improvements back into your code: Once their proprietary improvements have been imitated by their competitors, they'll grow tired of supporting the changes themselves. Once the advantage of a change is gone, it's easier for the vendor to make it public and to move on to other private improvements in search of a competitive advantage. The [X Window System License](#) for X11R6.4 at the Open Group site is the simplest example of a [BSD-type license](#).
- **Mozilla Public License.** This [license](#) enables you to protect your code while sharing it with others. Your software (and any improvements to it) are divided into two parts: the protected ("covered") portion and the contributed portion. If people want to modify and distribute the protected portion, they can, so long as they also distribute the source code for the modifications. If they want to change the software but keep their modifications private, they can distribute them without source code and must access the protected code only through its API. If they need to make changes to the API of the protected code, they can, as long as they distribute the source code to those changes. This license makes a neat bridge between the fully-open world of the GPL and the closed world of the BSD/X licenses.

## Proprietary opportunities with open-source software

Finally, there are opportunities to make money with proprietary binaries using your knowledge of open-source software. For instance, Loki Entertainment Software noticed that games were popular with Linux users, and that many popular games were not on Linux. They licensed the porting and distribution rights for Civilization, Railroad Tycoon, and other games; did the porting; and then licensed to Macmillan Publishers the exclusive distribution rights for these Linux games. Macmillan had experience bringing Red Hat Linux into wide distribution, and it fed the Loki games into this channel. Result: Now you can buy the Linux version of Civilization anywhere. Loki is working to expand its market by involving Yellow Dog Linux in porting the games to the PowerPC. All of this was done in a year.

## The new marketplace

As the Linux market grows, there will be more users than Linux experts. This will provide opportunities for the integration and support businesses described above. The value that open-source people place on personal independence makes it likely that most of these businesses will be small.

The open-source community has long been working and waiting for the day when Linux becomes widely used, and developers will be able to develop for a popular platform on which no vendor has a secret advantage. But the vendors in this new world may find it looks much like the world we now have. There will be no Very Large Software Company to scare ISVs into sticking to the niches, but any vendor with a horizontal product (that is, one in wide use), will face a host of competitors. Both the number of competitors and the availability of useful free products will help drive down application prices and profits. And vendors of widely-used proprietary products who keep their prices too high may find their products cloned! As a result, vendors will find they can make more money in the niches--where they are today.

One of the largest shifts in the software world (if it happens) will be the migration to Web-based applications. Linux is a natural for such an environment because of its remote-management capabilities. This predicted shift will involve not just a different technology, but a new economic model as well: Application software will become less of a product and more of a service. Competition between application vendors might take place along one of two different lines: as proprietary products, available from a single vendor or its multiple online service dealers; or as open-source products, maintained and improved by the community, but available from integrators or support organizations. These organizations will receive annual fees in return for installing and supporting the applications in their customer organizations.

There are no guarantees you'll make a fortune in open-source software, but there are increasing opportunities to make a living in this new world. Just keep your eyes open, find a need, and fill it.

## Resources

- "[Preparing for the intellectual-property offensive](#)," in *LinuxWorld* (November 1998)
- The [Open Source Initiative](#)
- The FreeBSD license, an example of a [BSD-type license](#)
- The X Window System license, another good [BSD-type license](#)
- The [Mozilla Public License](#)
- Register [copyrights](#)

## About the author

Donald K. Rosenberg is president of Stromian Technologies, an international consultancy for OEM software licensing problems, and publisher of the OEM Software Licensing Guide, the open source Software Licensing Page, and other online marketing resources at <http://www.stromian.com>. Dr. Rosenberg has more than 15 years of marketing experience and has worked with companies large and small in the United States and Europe, both in open source and more traditional software markets.

---

## What do you think of this article?

☐ Killer!   ☐ Good stuff   ☐ So-so; not bad   ☐ Needs work   ☐ Lame!

## Comments?

Submit feedback